CALCOLATORI ELETTRONICI – PROVA II – 12/7/2016 – II FACOLTA' DI INGEGNERIA (CESENA)

COGNOME	NOME	MATRICOLA	PARI/DISPARI	POSTO
			Non interessa	

TESTO - Tempo disponibile: 70 minuti

Prima domanda introduttiva al compito (Punti 3)

1. Facendo riferimento all'architettura del DLX vista a lezione si dica come i bit del registro IR (Instruction Register) della U.d.C. vengono collegati ai decoder del *register file* nelle istruzioni di tipo **ALU** con operandi sui registri e nelle istruzioni di tipo **ALU** con operando immediato.

Si vuole ora estendere l'ISA del DLX visto a lezione con istruzioni di accesso a uno STACK localizzato in memoria principale (cioè indirizzato tramite MAR).
A tal fine si assegna al registro **R30** la funzione di Stack Pointer (SP) e si adottano le seguenti convenzioni:

- Lo stack cresce verso gli indirizzi decrescenti
- Lo SP punta all'ultimo byte occupato (in caso di dati salvati in memoria in formato little endian, questo è il byte meno significativo dell'ultima word salvata sullo stack).

Inoltre si assume che SP (cioè R30) sia stato inizializzato con un valore multiplo di 4 (si assume cioè che gli elementi dello stack siano allineati).
Tra le istruzioni di accesso allo stack, si consideri ora la seguente:

- PUSH Rx, Ry

Questa istruzione salva due registri del register file sullo stack: prima Rx e poi Ry

L'istruzione **PUSH Rx, Ry** è quindi realizzata con le seguenti operazioni descritte in RTL:

M[R30-4] ← Rx
M[R30-8] ← Ry
R30 ← R30-8

2. Si ipotizzi che il codice operativo della PUSH sia **35H** e si mostri la *codifica della nuova istruzione* scegliendo uno dei 3 formati R, I, J delle istruzioni del DLX. Quanti operandi sorgente ha la nuova istruzione? Quanti operandi destinazione? Si indichi come il registro IR della U.d.C. dovrà essere collegato ai decoder del register file: si devono modificare queste connessioni rispetto a quelle individuate nella risposta alla domanda 1? Sono necessarie altre connessioni tra IR e datapath? **(punti 5)**

3. Con riferimento alla *codifica dell'istruzione* scelta e con riferimento al datapath del DLX sequenziale visto a lezione, si disegni ora il diagramma degli stati che controlla l'esecuzione dell'istruzione assegnata utilizzando il minor numero di stati possibile, inserendo anche gli stati necessari alle fasi di fetch e decodifica. **Quanti periodi di clock sono necessari per eseguire l'istruzione appena progettata nel caso in cui l'accesso alla memoria richieda 1 stato di wait? (Punti 15)**

4. Si chiede ora di scrivere nell'ISA del DLX "standard" la sequenza di istruzioni necessaria a eseguire la stessa funzione svolta dalla istruzione appena progettata e si calcoli il relativo speed up rispetto all'esecuzione della stessa funzione con il DLX standard. Si motivi questo speed up **(punti 5).**

5. Si chiede ora di indicare come si potrebbe modificare l'architettura del datapath, se si volesse realizzare l'istruzione di PUSH Rx, Ry senza utilizzare un registro del register file come Stack Pointer. In questo caso come e perchè si semplificherebbe il formato dell'istruzione? **(Punti 2).** Con quali istruzioni (e relativa codifica) si potrebbe pensare di inizializzare questo SP? **(Punti 3).**

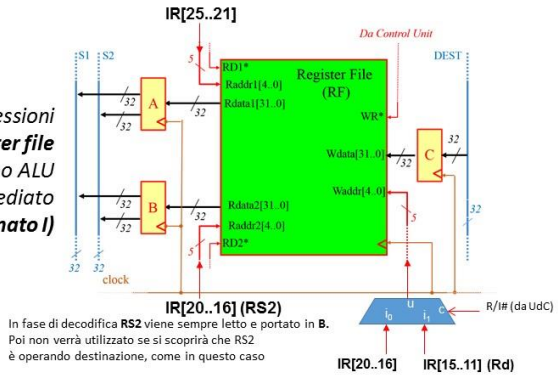
GUIDA ALLA SOLUZIONE

Quesito N.1

*Facendo riferimento all'architettura del DLX vista a lezione si dica come i bit del registro IR (Instruction Register) della U.d.C. vengono collegati ai decoder del register file nelle istruzioni di tipo **ALU** con operandi sui registri e nelle istruzioni di tipo **ALU** con operando immediato.*

Si veda la soluzione della prova II di giugno 2016 e si ritrovino le corrispondenze con la figura sottostante, che rappresenta la soluzione al quesito N. 1 nel caso delle istruzioni ALU con operando immediato (formato istruzione: I).

*Interconnessioni tra **IR** e **Register file** nelle istruzioni di tipo ALU con operando immediato **(formato I)***



Quesito N. 2 (punti 5)

*Si ipotizzi che il codice operativo della PUSH sia **35H** e si mostri la codifica della nuova istruzione scegliendo uno dei 3 formati R, I, J delle istruzioni del DLX. Quanti operandi sorgente ha la nuova istruzione? Quanti operandi destinazione? Si indichi come il registro IR della U.d.C. dovrà essere collegato ai decoder del register file: si devono modificare queste connessioni rispetto a quelle individuate nella risposta alla domanda 1? Sono necessarie altre connessioni tra IR e datapath?*

Scegliamo il formato R e decidiamo di inserire Rx e Ry in RS1 e RS2 e inseriamo la costante 30 (11110) nel campo Rd (che finirà in IR[15..11]).

IR →	IR[31..26]	IR[25..21]	IR[20..16]	IR[15..11]	IR[10..0]
Formato	**Cop**	**RS1**	**RS2**	**Rd**	**11bit spare**
R	110101 (35H)	Rx	Ry	11110	000 0000 0100

L'istruzione assegnata *PUSH Rx, Ry* ha tre registri sorgente (*Rx, Ry* e R30) e un registro destinazione (R30). Quindi con il formato suindicato, Rx e Ry verranno portati in A e B durante la fase di decodifica dell'istruzione, mentre per leggere anche R30 sarà necessario veicolare al decoder di una delle due porte di lettura del register file anche Rd; questo implica l'aggiunta di un MUX che consenta di portare anche il campo Rd a RAddr1[4..0] o a RAddr2[4..0] in modo simile a come il mux azzurro della figura soprastante porta RS2 o Rd a Waddr[4..0]. *Scegliamo di inserire questo MUX (da 5 bit, a 2 vie) sugli ingressi R1Addr[4..0] in modo da poter controllare la porta di lettura 1 del RF con RS1 oppure con Rd.*
Solo così sarà legittimo inserire nel dds la microistruzione: **A ← Rd.**
Infine inseriamo nei bit [10..0] (liberi) la costante 4 che dovremo sottrarre due volte a R30.

Quesito n. 3 (Punti 15)

Con riferimento alla codifica dell'istruzione scelta e con riferimento al datapath del DLX sequenziale visto a lezione, si disegni ora il diagramma degli stati che controlla l'esecuzione dell'istruzione assegnata utilizzando il minor numero di stati possibile, inserendo anche gli stati necessari alle fasi di fetch e decodifica. Quanti periodi di clock sono necessari per eseguire l'istruzione appena progettata nel caso in cui l'accesso alla memoria richieda 1 stato di wait?

Nella fase di decodifica Rx e Ry vengono trasferiti su A e B
La prima microistruzione a valle della decodifica porta Rx su MDR e porta R30 in A
La successiva inizializza MAR con R30-4
Poi si salva Rx sullo stack
Segue l'aggiornamento di R30 e il salvataggio di Ry sullo stack
Si noti la concorrenza tra:
l'aggiornamento di R30 e
il salvataggio di Ry sullo stack.
L'istruzione esegue 3 cicli di bus: 1 di lettura e 2 di scrittura in memoria.
In base al numero di stati di wait (n), il CPI dell'istruzione sarà: 8 + 3*n
Nel caso in cui la memoria richieda 1 stato di wait, si ha:
CPI = 11
Sarà conveniente introdurre la nuova istruzione nell'ISA se questa istruzione sarà molto frequente nel codice eseguito da questa CPU.

Quesito n. 4 *(punti 5)*

Si chiede ora di scrivere nell'ISA del DLX "standard" la sequenza di istruzioni necessaria a eseguire la stessa funzione svolta dalla istruzione appena progettata e si calcoli il relativo speed up rispetto all'esecuzione della stessa funzione con il DLX standard. Si motivi questo speed up.

PUSH Rx, Ry:

SW -4 (R30), Rx (7 Tck con uno stato di wait per ogni accesso in mem)
SW -8 (R30), Ry
SUBI R30, R30, 8 (6 Tck con uno stato di wait per ogni accesso in mem)

Sono dunque necessarie tre istruzioni "native" DLX per realizzare l'istruzione assegnata.
Lo speed up dell'istruzione microprogrammata è dato dal rapporto tra il CPUtime del frammento di codice assembler soprastante e il valore di CPI dell'istruzione microprogrammata.
Quindi:
SPEEDUP: 20/11 = 1,81 (+81%)
Il risparmio di tempo (cioè la riduzione del tempo di esecuzione della PUSH) è dovuto principalmente al fatto che per eseguire una PUSH Rx,Ry senza che l'istruzione sia definita dall'ISA sono necessarie 3 fasi di Fetch e Decode, che da sole richiedono 9 Tck invece di 3Tck. La parte restante del risparmio deriva dall'esecuzione concorrente dell'aggiornamento di R30 e del salvataggio sullo stack di Ry.

Quesito 5.
Si chiede ora di indicare come si potrebbe modificare l'architettura del datapath, se si volesse realizzare l'istruzione di PUSH Rx, Ry senza utilizzare un registro del register file come Stack Pointer. In questo caso come e perchè si semplificherebbe il formato dell'istruzione? ***(Punti 2).*** *Con quali istruzioni (e relativa codifica) si potrebbe pensare di inizializzare questo SP?* ***(Punti 3).***

Per disporre di uno stack è necessario avere uno stack pointer (SP). Quindi se non possiamo riservarci un registro del RF per svolgere la funzione di SP, potremmo aggiungere al data path un ulteriore registro da 32 bit, che si affianchi agli altri; potremmo chiamare questo registro SP o TEMP2 per differenziarlo dagli altri.
L'aggiunta di un registro al data path implica l'aggiunta anche di 3 segnali di controllo: i due output enable (OE1# e OE2#) e il write enable (WE#).
Il formato dell'istruzione si semplificherebbe rispetto a quello proposto nella risposta al quesito 2 in quanto non sarebbe necessario inserire R30 nei campi dell'istruzione che selezionano registri del RF. Inoltre non sarebbe necessario aggiungere il MUX di controllo dei segnali RAddr1[4..0] (si veda la risposta al quesito 2).

Si noti che l'aggiunta di uno SP e delle istruzioni di PUSH e POP non sono sufficienti per utilizzare lo stack. E' infatti necessario disporre anche di due istruzioni che consentano di inizializzare e leggere SP. Queste ulteriori istruzioni potrebbero ad esempio trasferire il valore di un registro del RF a SP e viceversa:

- Carica SP: **SP←Ri**
- Leggi SP: **Ri←SP**
-

Queste due istruzioni potrebbero essere codificate in formato R con un loro specifico codice operativo.
I bit significativi di queste istruzioni sarebbero ovviamente 11.